PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2003-044272
(43)Date of publication of application : 14.02.2003

(51)Int.CI. G06F 9/30
G06F 9/38
G06F 15/16

(21)Application number : 2001-235388
(22)Date of filing : 02.08.2001

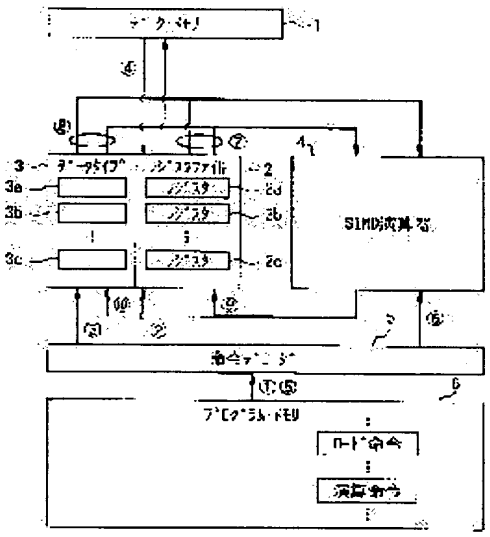
(71)Applicant : SEIKO EPSON CORP
(72)Inventor : ISOMURA MASAICHI

(54) PROCESSOR, DATA TYPE SPECIFYING METHOD, DATA LOADING METHOD, COMPUTING METHOD, DATA TYPE UPDATING METHOD AND INSTRUCTION PROGRAM

(57)Abstract:

PROBLEM TO BE SOLVED: To reduce the number of SIMD type instructions without damaging the diversity of arithmetic functions.

SOLUTION: An instruction decoder 5 discriminates a data type from a loading instruction and stores the data type in data type storage area 3a to 3c corresponding to registers 2a to 2c storing data. An SIMD computing element 4 determines the type of the data to be a computation object on the basis of the data stored in the data type storage areas 3a to 3c.



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]
[Claim 1] The data memory which memorizes data, and the input register which loads the data specified by the load instruction from said data memory, A data-type assignment means to specify the data type of the data specified by said load instruction, The data-type storage region which said input register is made to correspond and memorizes the data type specified with said data-type assignment means, The processor characterized by having the computing element which calculates the data loaded to said input register based on the data type memorized in said data-type storage region.
[Claim 2] The data memory which memorizes data, and the input register which loads the data specified by the load instruction from said data memory, A data-type assignment means to specify the data type of the data specified by said load instruction, The 1st data-type storage region which said input register is made to correspond and memorizes the data type specified with said data-type assignment means, The computing element which calculates the data loaded to said input register based on the data type memorized in said data-type storage region, The processor characterized by having the output register which memorizes the result of an operation of the data based on said computing element, and the 2nd data-type storage region which the data type corresponding to the result of an operation of said data is made to correspond to said output register, and memorizes it.
[Claim 3] The processor characterized by having an operation means to calculate the input register which memorizes the data with which data types differ, and the data with which said data types differ, and the output register which memorizes the result of an operation by said operation means.
[Claim 4] The data-type specification method characterized by specifying the data type of a SIMD mold instruction by the load instruction.
[Claim 5] Said load instruction is a data-type specification method according to claim 4 characterized by being prepared for every data type.
[Claim 6] The data load approach characterized by having the step which loads the data specified by the load instruction to a register, and the step which said register is made to correspond and memorizes the data type corresponding to said loaded data.
[Claim 7] The operation approach characterized by having the step which loads the data specified by the load instruction to an input register, the step which said input register is made to correspond and memorizes the data type corresponding to said loaded data, the step which inputs into a computing element the data loaded to said input register with said data type, and the step which calculates said data based on said data type.
[Claim 8] The step which loads the data specified by the load instruction to an input register, The step which said input register is made to correspond and memorizes the data type corresponding to said loaded data, The step which inputs into a computing element the data loaded to said input register with said data type, The step which calculates said data based on said data type, The renewal approach of a data type characterized by having the step which memorizes the result of an operation of said data to an output register, and the step which the data type corresponding to the result of an operation of said data is made to correspond to said output register, and memorizes it.
[Claim 9] The instruction program characterized by making a computer perform the step which specifies the data type of a SIMD mold instruction by the load instruction, the step which loads the data specified by said load instruction to a register, and the step which said register is made to correspond and memorizes the data type specified by said load instruction.
[Claim 10] The step which specifies the data type of a SIMD mold instruction by the load instruction, The step which loads the data specified by said load instruction to a register, The step which said register is

made to correspond and memorizes the data type specified by said load instruction, The instruction program characterized by making a computer perform the step which inputs into a computing element the data loaded to said register with said data type, and the step which makes data processing perform to said computing element based on said data type.
[Claim 11] The step which specifies the data type of a SIMD mold instruction by the load instruction, The step which loads the data specified by said load instruction to an input register, The step which said input register is made to correspond and memorizes the data type specified by said load instruction, The step which inputs into a computing element the data loaded to said input register with said data type, The step which makes data processing perform to said computing element based on said data type, The instruction program characterized by making a computer perform the step which memorizes said data-processing result to an output register, and the step which the data type corresponding to said data-processing result is made to correspond to said output register, and memorizes it.

[Translation done.]

DETAILED DESCRIPTION

[Detailed Description of the Invention]
[0001]
[Field of the Invention] Especially this invention is applied to a SIMD (Single Instruction Multi Data) mold processor about a processor, a data-type specification method, the data load approach, the operation approach, the renewal approach of a data type, and an instruction program, and is suitable.
[0002]
[Description of the Prior Art] In the conventional processor, in order to raise data processing capacity, there are some which adopted SIMD mold data format. In this SIMD mold processor, the SIMD mold operation instruction for making two or more data process with one instruction is prepared, and the SIMD mold parallel operation machine for executing this SIMD mold operation instruction is formed.
[0003] Ultra which has the Pentium (trademark) processor which adopted the MMX technology of for example, U.S. Intel, and a VIS instruction of the U.S. Sun Microsystem company as such a SIMD mold processor There is a Sparc processor etc. Drawing 6 is drawing showing the data type of the conventional MMX technology. In drawing 6 , four kinds of data format, pack DOBAITO (eight 8 bit data) of drawing 6 (a), pack DOWADO (four 16 bit data) of drawing 6 (b), the PAKKUDO double word (two 32 bit data) of drawing 6 (c), and the quad WORD (one 64 bit data) of drawing 6 (d), is prepared as a data format stored in a 64-bit MMX register.
[0004] Thus, in a SIMD mold processor, two or more data format stored in a register exists. For this reason, in order to perform the SIMD operation of such data, the SIMD mold instruction is prepared for every data type. For example, by MMX technology, in order to perform addition, the add instruction is prepared for three kinds of every data types, pack DOBAITO of drawing 6 (a), pack DOWADO of drawing 6 (b), and the PAKKUDO double word of drawing 6 (c). Moreover, if the addition with SACHURESHON which prevents the underflow of the result of an operation and overflow is included, seven kinds of instructions are prepared only by the add instruction. Furthermore, it is prepared for every data type also about the instruction of not only addition but subtraction, multiplication, a shift, etc.
[0005] For this reason, much instructions were defined by the conventional SIMD mold processor in order to execute a SIMD mold instruction.
[0006]
[Problem(s) to be Solved by the Invention] However, since many numbers of bits needed to be assigned to the operand for distinguishing these instructions when the number of instructions increased, the whole instruction code increased and there was a problem that the code effectiveness of a program fell. Drawing 7 is drawing showing an example of the conventional instruction code.
[0007] In drawing 7 , instruction code consists of 32 bits and can specify Operand OP, an output register rt, and input registers rs and rd by this instruction code. Here, if the number of instructions increases, many numbers of bits will be needed for Operand OP, and the whole instruction code will increase.
[0008] On the other hand, although there is also the approach of offsetting the increment of the number of bits of Operand OP by reducing the number of bits assigned to an output register rt and input registers rs and rd in order to prevent decline in the code effectiveness of a program, the degree of freedom of a register select is restrained by this approach. Then, the purpose of this invention is offering the processor which can reduce the number of instructions of a SIMD mold instruction, a data-type specification method, the data load approach, the operation approach, the renewal approach of a data type, and an instruction program, without spoiling the versatility of a calculation function.
[0009]
[Means for Solving the Problem] In order to solve the technical problem mentioned above, according to

the processor according to claim 1 The data memory which memorizes data, and the input register which loads the data specified by the load instruction from said data memory, A data-type assignment means to specify the data type of the data specified by said load instruction, The data-type storage region which said input register is made to correspond and memorizes the data type specified with said data-type assignment means, It is characterized by having the computing element which calculates the data loaded to said input register based on the data type memorized in said data-type storage region.
[0010] Thereby, only by preparing a load instruction for every data type, since it becomes possible to distinguish a data type and it becomes unnecessary to prepare the instruction corresponding to a data type for every operation classification, it becomes possible to reduce the number of instructions of a SIMD mold instruction. Furthermore, a computing element only refers to a data-type storage region, becomes possible [distinguishing the data type of the data used as the candidate for an operation], and it becomes possible [performing various operations with the small number of instructions] from it becoming possible to perform the operation according to a data type, without giving a different instruction for every data type.
[0011] Moreover, the data memory which memorizes data according to the processor according to claim 2, The input register which loads the data specified by the load instruction from said data memory, A data-type assignment means to specify the data type of the data specified by said load instruction, The 1st data-type storage region which said input register is made to correspond and memorizes the data type specified with said data-type assignment means, The computing element which calculates the data loaded to said input register based on the data type memorized in said data-type storage region, It is characterized by having the output register which memorizes the result of an operation of the data based on said computing element, and the 2nd data-type storage region which the data type corresponding to the result of an operation of said data is made to correspond to said output register, and memorizes it.
[0012] When this becomes possible to update a data type according to the result of an operation and a data type changes by data processing, whenever it can maintain the adjustment of a data type and operation instruction is executed, the need of respecifying a data type can be abolished. Moreover, according to the processor according to claim 3, it is characterized by having an operation means to calculate the input register which memorizes the data with which data types differ, and the data with which said data types differ, and the output register which memorizes the result of an operation by said operation means.
[0013] It becomes possible to increase the variation of a SIMD operation, without this becoming possible [defining an operation based on the combination of a data type], and making the number of instructions increase. Moreover, according to the data-type specification method according to claim 4, it is characterized by specifying the data type of a SIMD mold instruction by the load instruction.
[0014] Since it becomes possible to distinguish a data type and it becomes unnecessary to prepare the instruction corresponding to a data type for every operation classification by this in case data are loaded to a register, it becomes possible to reduce the number of instructions of a SIMD mold instruction. Moreover, according to the data-type specification method according to claim 5, it is characterized by preparing said load instruction for every data type.
[0015] Thereby, without preparing the instruction corresponding to a data type for every operation classification, it becomes possible to distinguish a data type and it becomes possible to reduce the number of instructions of a SIMD mold instruction. Moreover, according to the data load approach according to claim 6, it is characterized by having the step which loads the data specified by the load instruction to a register, and the step which said register is made to correspond and memorizes the data type corresponding to said loaded data.
[0016] In case this incorporates the data loaded to the register, the data type can also be incorporated to coincidence, and since it becomes possible to perform the operation according to a data type only by giving a single instruction, it becomes possible to reduce the number of instructions of a SIMD mold instruction. Moreover, it is characterized by to have the step which loads the data specified by the load instruction to an input register, the step which said input register is made to correspond and memorizes the data type corresponding to said loaded data, the step which inputs into a computing element the data loaded to said input register with said data type, and the step which calculates said data based on said data type according to the operation approach according to claim 7.
[0017] It becomes possible only by this preparing a load instruction for every data type to distinguish a data type, and while becoming possible to reduce the number of instructions of a SIMD mold instruction, it becomes possible from it becoming possible to perform the operation according to a data type only by giving a single instruction to perform various operations with the small number of instructions.
[0018] Moreover, the step which loads the data specified by the load instruction to an input register

according to the renewal approach of a data type according to claim 8, The step which said input register is made to correspond and memorizes the data type corresponding to said loaded data, The step which inputs into a computing element the data loaded to said input register with said data type, It is characterized by having the step which calculates said data based on said data type, the step which memorizes the result of an operation of said data to an output register, and the step which the data type corresponding to the result of an operation of said data is made to correspond to said output register, and memorizes it.

[0019] This becomes possible [updating a data type according to the result of an operation], and when a data type changes by data processing, the adjustment of a data type can be maintained. Moreover, according to the instruction program according to claim 9, it is characterized by making a computer perform the step which specifies the data type of a SIMD mold instruction by the load instruction, the step which loads the data specified by said load instruction to a register, and the step which said register is made to correspond and memorizes the data type specified by said load instruction.

[0020] Since it becomes possible to distinguish a data type and it becomes unnecessary to prepare the instruction corresponding to a data type for every operation classification by this in case data are loaded to a register, it becomes possible to reduce the number of instructions of a SIMD mold instruction. Moreover, the step which specifies the data type of a SIMD mold instruction by the load instruction according to the instruction program according to claim 10, The step which loads the data specified by said load instruction to a register, The step which said register is made to correspond and memorizes the data type specified by said load instruction, It is characterized by making a computer perform the step which inputs into a computing element the data loaded to said register with said data type, and the step which makes data processing perform to said computing element based on said data type.

[0021] It becomes possible only by this preparing a load instruction for every data type to distinguish a data type, and while becoming possible to reduce the number of instructions of a SIMD mold instruction, it becomes possible from it becoming possible to perform the operation according to a data type only by giving a single instruction to perform various operations with the small number of instructions.

[0022] Moreover, the step which specifies the data type of a SIMD mold instruction by the load instruction according to the instruction program according to claim 11, The step which loads the data specified by said load instruction to an input register, The step which said input register is made to correspond and memorizes the data type specified by said load instruction, The step which inputs into a computing element the data loaded to said input register with said data type, The step which makes data processing perform to said computing element based on said data type, It is characterized by making a computer perform the step which memorizes said data-processing result to an output register, and the step which the data type corresponding to said data-processing result is made to correspond to said output register, and memorizes it.

[0023] This becomes possible [updating a data type according to the result of an operation], and when a data type changes by data processing, the adjustment of a data type can be maintained.

[0024]

[Embodiment of the Invention] Hereafter, the processor concerning the operation gestalt of this invention is explained, referring to a drawing. Drawing 1 is the block diagram showing the outline configuration of the processor concerning the 1st operation gestalt of this invention.

[0025] In drawing 1 , data are stored in data memory 1, according to a load instruction, data are loaded to a register file 2, or the data of a register file 2 are stored. While Registers 2a-2c are formed, the data-type storage region 3 is established in the register file 2. Here, the data-type storage region 3 is established in every register 2a - 2c, and the data-type storage regions 3a-3c are formed corresponding to these registers 2a-2c.

[0026] Drawing 2 is drawing showing the example of a configuration of the registers 2a-2c concerning the 1st operation gestalt of this invention. In drawing 2 , the 32-bit data storage area 11 is established in Registers 2a-2c, and the data-type storage region 12 for 2 bits is added to the data storage area 11.

[0027] Here, as a data type, four kinds, 4 8 bits packed data, 2 16 bits packed data, 32-bit data, and floating a small number of point formal data, are established, and "11" is assigned to 4 8-bit packed data to "10" and floating a small number of point formal data as discernment data for identifying a data type at "00" and 2 16-bit packed data at "01" and 32-bit data.

[0028] When 4 8-bit packed data are stored in a data storage area 11, and in the data-type storage region 12 When "00" is stored corresponding to the data storage area 11 and 2 packed data which are 16 bits are stored in a data storage area 11, in the data-type storage region 12 When "01" is stored corresponding to the data storage area 11 and the data which are 32 bits are stored in a data storage area 11, in the data-type

storage region 12 When "10" is stored corresponding to the data storage area 11 and floating a small number of point formal data are stored in a data storage area 11, "11" is stored in the data-type storage region 12 corresponding to the data storage area 11.
[0029] The SIMD computing element 4 is connected to the register file 2, the data-type storage region 3, and the instruction decoder 5. And the data memorized by the register file 2 are read with the data type memorized in the data-type storage region 3, and the operation according to the data type is performed. And while outputting the result of an operation to a register file 2, the data type corresponding to the result of an operation is outputted to the data-type storage region 3.
[0030] An instruction decoder 5 generates the control signal which controls a register file 2, the data-type storage region 3, and the SIMD computing element 4 based on the instruction code read from program memory 6. Instruction code is stored in program memory 6, and instruction code is outputted to an instruction decoder 5. Here, a load instruction is prepared for every data type.
[0031] Drawing 3 is drawing showing an example of the instruction concerning the 1st operation gestalt of this invention. In drawing 3 , as for a load instruction, four kinds of instructions LDPB, LDPW, and LDDW and LDFS are prepared corresponding to 4 [8-bit] of drawing 2 packed data, 2 16 bits packed data, 32-bit data, and floating a small number of point formal data. Moreover, operation instruction, such as addition and subtraction, is prepared common to a different data type.
[0032] Next, actuation of the processor of drawing 1 is explained. If a load instruction is read from program memory 6, (**) and an instruction decoder 5 will decode a load instruction. And the data read from data memory 1 are stored in the registers 2a-2c specified by (**) and the load instruction (**). Here, a load instruction is prepared for every data type, and an instruction decoder 5 distinguishes a data type from a load instruction. And the data-type storage regions 3a-3c corresponding to the registers 2a-2c with which data were memorized are made to memorize the data type (**).
[0033] For example, the mnemonic of a load instruction is LDPB of drawing 3 , by this load instruction, supposing register 2a in a register file 2 is specified as an input register, by decoding this mnemonic LDPB, the data used as the candidate for a load will distinguish an instruction decoder 5 from 4 8-bit packed data of drawing 2 , and it will acquire "00" as discernment data of that data type.
[0034] And an instruction decoder 5 makes data-type storage region 3a prepared corresponding to this register 2a memorize "00" while it makes these 4 8-bit packed data read from data memory 1 and making register 2a of a register file 2 memorize it. This becomes possible to distinguish the data type of the data memorized by Registers 2a-2c only by preparing a load instruction for every data type.
[0035] Next, if operation instruction is read from program memory 6, (**) and an instruction decoder 5 will decode the operation instruction, and will output a control signal to the SIMD computing element 4 (**). If a control signal is outputted from an instruction decoder 5, the SIMD computing element 4 will incorporate a data type from the data-type storage regions 3a-3c corresponding to (**) and its registers 2a-2c while incorporating data from the registers 2a-2c specified there (**). And while performing the SIMD operation of the data according to the data type and storing the result of an operation in the specified registers 2a-2c, (**) and the data-type storage regions 3a-3c corresponding to the registers 2a-2c of the output destination change are made to memorize a data type (round head 10).
[0036] For example, an add instruction shall be read from program memory 6, register 2a in a register file 2 and 2b shall be specified as an input register by this add instruction, and register 2c shall be specified as an output register. In this case, an instruction decoder 5 specifies register 2a and 2b as an input register, and specifies register 2c as an output register while it performs addition directions to the SIMD computing element 4.
[0037] The SIMD computing element 4 reads a data type from the data-type storage regions 3a and 3b corresponding to register 2a and 2b while reading data from register 2a and 2b, if register 2a and 2b are specified as an input register. And the SIMD computing element 4 outputs a data type to data-type storage region 3c corresponding to that register 2c while it adds the data of register 2a and 2b and outputs that addition result to register 2c according to this data type.
[0038] Here, operation instruction is prepared common to a data type, and an instruction decoder 5 sends the signal which directs an input register and an output register to a register file 2 while sending only directions of addition, multiplication, etc. to the SIMD computing element 4, if operation instruction is sent from program memory 6. And the SIMD computing element 4 judges the data type of the data used as the candidate for an operation based on the data memorized in the data-type storage regions 3a-3c rather than judges the data type of the data used as the candidate for an operation based on directions of the operation from an instruction decoder 5.
[0039] It becomes possible for it to become unnecessary to prepare the instruction corresponding to a data

type for every operation classification, such as addition, subtraction, and multiplication, and to reduce the number of instructions of a SIMD mold instruction by this. In addition, you may make it it not only to specify the data type memorized in the data-type storage region 3 by the load instruction, but update it based on the result of an operation by the SIMD computing element 4.
[0040] Drawing 4 is drawing showing the updating approach of the data type concerning the 2nd operation gestalt of this invention. In drawing 4 , while data-type discernment data "01" are memorized by data-type storage region 12a while 2 16-bit packed data a0 and a1 are memorized by data storage area 11a, and 2 16-bit packed data b0 and b1 are memorized by data storage area 11b, data-type discernment data "01" shall be memorized by data-type storage region 12b, and the multiplication of these data shall be performed.
[0041] In this case, while 32-bit data are generated by the multiplication of a0 and b0 and multiplication result a0xb0 [those 32 bits] is memorized by data storage area 11c, 32-bit data are generated by the multiplication of a1 and b1, and multiplication result a1xb1 [those 32 bits] is memorized at 11d of data storage areas. Here, since the data with which the data memorized in data storage areas 11a and 11b are memorized in 2 16-bit packed data and data storage areas 11c and 11d are 32-bit data, data types differ before and after the operation.
[0042] For this reason, the SIMD computing element 4 updates the data-type discernment data "01" memorized in the data-type storage regions 12a and 12b, and memorizes the data-type discernment data "10" corresponding to 32-bit data in the data-type storage regions 12c and 12d. Without giving a different instruction for every data type by this, when a data type changes according to the result of an operation, it becomes possible to maintain the adjustment of a data type, and it becomes possible only by referring to the data-type storage regions 12c and 12d to perform subsequent processing appropriately.
[0043] Drawing 5 is drawing showing the operation approach concerning the 3rd operation gestalt of this invention. In drawing 5 , while the 32-bit data a are memorized by data storage area 21a, 4 8-bit packed data b0, b1, b2, and b3 are memorized by data storage area 21b. Addition with the 32 bits data a and 4 8-bit packed data b0, b1, b2, and b3 is defined as addition with 8 bits and 4 8-bit packed data b0, b1, b2, and b3 of the 32-bit data a, and it enables it to add these data with the SIMD computing element 4 of drawing 1 here.
[0044] And if an add instruction is outputted to the SIMD computing element 4, the SIMD computing element 4 will perform addition with the data a of data storage area 21a, and the data b0, b1, b2, and b3 of data storage area 21b according to the above-mentioned definition. and -- ******** -- as an addition result of data types, if a+b0, a+b1, a+b1, and a+b1 are obtained, these addition results will be memorized to data storage area 21c as 4 8-bit packed data.
[0045] Thereby, without increasing the number of instructions, BARIESHON of an operation can be increased and it becomes possible to raise the code effectiveness of a program.
[0046]
[Effect of the Invention] It becomes possible to reduce the number of instructions of a SIMD mold instruction, without according to this invention, becoming possible to distinguish a data type and spoiling the versatility of data processing only by preparing a load instruction for every data type, as explained above.

[Translation done.]

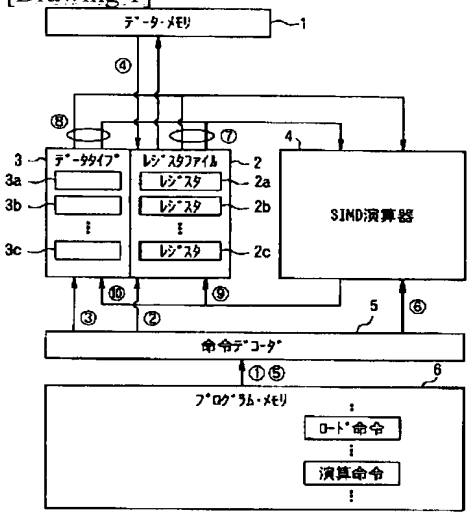
* NOTICES *

1.This document has been translated by computer. So the translation may not reflect the original precisely.

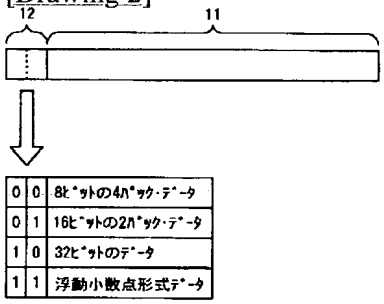
2.**** shows the word which can not be translated.
3.In the drawings, any words are not translated.

DRAWINGS

[Drawing 1]



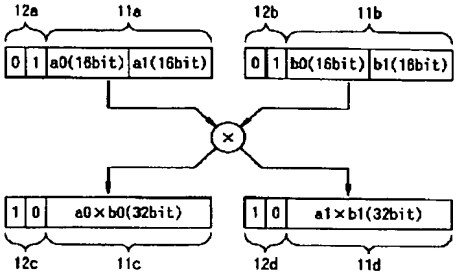
[Drawing 2]



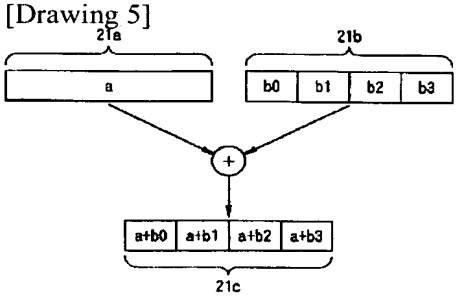
[Drawing 3]

機能	ニーモニック
ﾛｰﾄﾞ	LDPB,LDPW,LDDW,LDFS
加算	ADD
減算	SUB
ｻﾁｭﾚｰｼｮﾝ付き加算	ADDS
ｻﾁｭﾚｰｼｮﾝ付き減算	SUBS
⋮	⋮

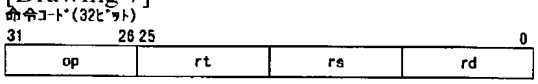
[Drawing 4]



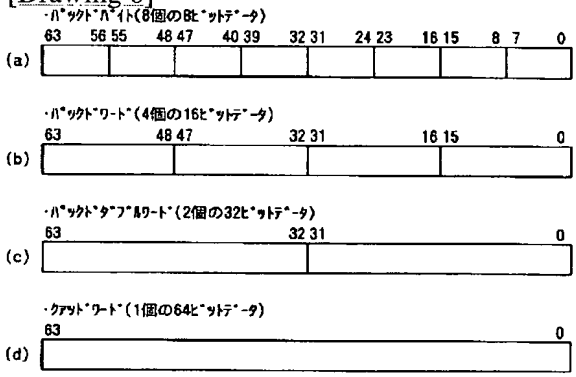
[Drawing 5]



[Drawing 7]



[Drawing 6]



[Translation done.]

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2003－44272
(P2003－44272A)
(43)公開日　平成15年２月14日(2003.2.14)

(51)Int.Cl.⁷		識別記号	FI		テーマコード(参考)	
Ｇ０６Ｆ	9/30	３１０	Ｇ０６Ｆ	9/30	３１０Ａ	５Ｂ０１３
		３５０			３５０Ｂ	５Ｂ０３３
	9/38	３７０		9/38	３７０Ａ	５Ｂ０４５
	15/16	６１０		15/16	６１０Ａ	

審査請求　未請求　請求項の数11　ＯＬ　（全８頁）

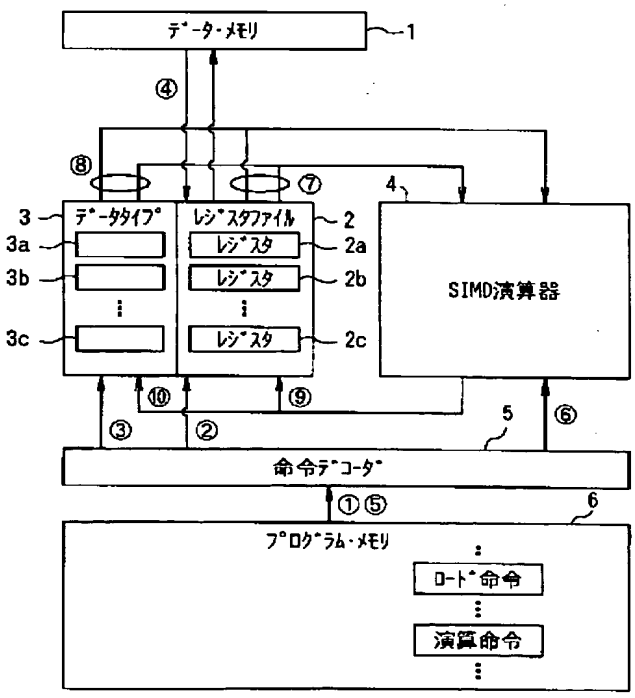
(21)出願番号　特願2001－235388(P2001－235388)
(22)出願日　平成13年８月２日(2001.8.2)

(71)出願人　000002369
セイコーエプソン株式会社
東京都新宿区西新宿２丁目４番１号
(72)発明者　礒村　政一
長野県諏訪市大和３丁目３番５号　セイコーエプソン株式会社内
(74)代理人　100095728
弁理士　上柳　雅誉　（外２名）

Ｆターム(参考)　5B013 DD01 DD04
5B033 AA06 BA01 DC07
5B045 GG14

(54)【発明の名称】　プロセッサ、データタイプ指定方法、データロード方法、演算方法、データタイプ更新方法および命令プログラム

(57)【要約】
【課題】　演算機能の多様性を損なうことなく、ＳＩＭＤ型命令の命令数を減らす。
【解決手段】　命令デコーダ５は、ロード命令からデータタイプを判別し、データが記憶されたレジスタ２ａ～２ｃに対応するデータタイプ記憶領域３ａ～３ｃに、そのデータタイプを記憶させ、ＳＩＭＤ演算器４は、データタイプ記憶領域３ａ～３ｃに記憶されているデータに基づいて、演算対象となるデータのデータタイプを判断する。



【特許請求の範囲】

【請求項１】　データを記憶するデータメモリと、
ロード命令で特定されるデータを前記データメモリからロードする入力レジスタと、
前記ロード命令で特定されるデータのデータタイプを指定するデータタイプ指定手段と、
前記データタイプ指定手段で指定されたデータタイプを、前記入力レジスタに対応させて記憶するデータタイプ記憶領域と、
前記データタイプ記憶領域に記憶されたデータタイプに基づいて、前記入力レジスタにロードされたデータの演算を行う演算器とを備えることを特徴とするプロセッサ。

【請求項２】　データを記憶するデータメモリと、
ロード命令で特定されるデータを前記データメモリからロードする入力レジスタと、
前記ロード命令で特定されるデータのデータタイプを指定するデータタイプ指定手段と、
前記データタイプ指定手段で指定されたデータタイプを、前記入力レジスタに対応させて記憶する第１データタイプ記憶領域と、
前記データタイプ記憶領域に記憶されたデータタイプに基づいて、前記入力レジスタにロードされたデータの演算を行う演算器と、
前記演算器によるデータの演算結果を記憶する出力レジスタと、
前記データの演算結果に対応したデータタイプを、前記出力レジスタに対応させて記憶する第２データタイプ記憶領域とを備えることを特徴とするプロセッサ。

【請求項３】　データタイプの異なるデータを記憶する入力レジスタと、
前記データタイプの異なるデータの演算を行う演算手段と、
前記演算手段による演算結果を記憶する出力レジスタとを備えることを特徴とするプロセッサ。

【請求項４】　ＳＩＭＤ型命令のデータタイプをロード命令により指定することを特徴とするデータタイプ指定方法。

【請求項５】　前記ロード命令はデータタイプごとに用意されていることを特徴とする請求項４記載のデータタイプ指定方法。

【請求項６】　ロード命令で特定されるデータをレジスタにロードするステップと、
前記ロードされたデータに対応するデータタイプを、前記レジスタに対応させて記憶するステップとを備えることを特徴とするデータロード方法。

【請求項７】　ロード命令で特定されるデータを入力レジスタにロードするステップと、
前記ロードされたデータに対応するデータタイプを、前記入力レジスタに対応させて記憶するステップと、

前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、
前記データタイプに基づいて、前記データの演算を行うステップとを備えることを特徴とする演算方法。

【請求項８】　ロード命令で特定されるデータを入力レジスタにロードするステップと、
前記ロードされたデータに対応するデータタイプを、前記入力レジスタに対応させて記憶するステップと、
前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、
前記データタイプに基づいて、前記データの演算を行うステップと、
前記データの演算結果を出力レジスタに記憶するステップと、
前記データの演算結果に対応したデータタイプを、前記出力レジスタに対応させて記憶するステップとを備えることを特徴とするデータタイプ更新方法。

【請求項９】　ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、
前記ロード命令で特定されるデータをレジスタにロードするステップと、
前記ロード命令で指定されるデータタイプを、前記レジスタに対応させて記憶するステップとをコンピュータに実行させることを特徴とする命令プログラム。

【請求項１０】　ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、
前記ロード命令で特定されるデータをレジスタにロードするステップと、
前記ロード命令で指定されるデータタイプを、前記レジスタに対応させて記憶するステップと、
前記レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、
前記データタイプに基づいて、前記演算器に演算処理を行わせるステップとをコンピュータに実行させることを特徴とする命令プログラム。

【請求項１１】　ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、
前記ロード命令で特定されるデータを入力レジスタにロードするステップと、
前記ロード命令で指定されるデータタイプを、前記入力レジスタに対応させて記憶するステップと、
前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、
前記データタイプに基づいて、前記演算器に演算処理を行わせるステップと、
前記演算処理結果を出力レジスタに記憶するステップと、
前記演算処理結果に対応したデータタイプを、前記出力レジスタに対応させて記憶するステップとをコンピュータに実行させることを特徴とする命令プログラム。

3

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、プロセッサ、データタイプ指定方法、データロード方法、演算方法、データタイプ更新方法および命令プログラムに関し、特に、ＳＩＭＤ（Ｓｉｎｇｌｅ　Ｉｎｓｔｒｕｃｔｉｏｎ　Ｍｕｌｔｉ　Ｄａｔａ）型プロセッサに適用して好適なものである。

【０００２】

【従来の技術】従来のプロセッサでは、データ処理能力を向上させるために、ＳＩＭＤ型データ形式を採用したものがある。このＳＩＭＤ型プロセッサでは、複数のデータの処理を１命令で行わせるためのＳＩＭＤ型演算命令が用意され、このＳＩＭＤ型演算命令を実行するためのＳＩＭＤ型並列演算器が設けられている。

【０００３】このようなＳＩＭＤ型プロセッサとして、例えば、米国インテル社のMMXテクノロジを採用したペンティアム（登録商標）・プロセッサや、米国サン・マイクロシステム社のＶＩＳ命令を持つＵｌｔｒａ　Ｓｐａｒｃプロセッサなどがある。図６は、従来のMMXテクノロジのデータタイプを示す図である。図６において、例えば、６４ビットのMMXレジスタに格納されるデータ形式として、図６（ａ）のパックドバイト（８個の８ビットデータ）、図６（ｂ）のパックドワード（４個の１６ビットデータ）、図６（ｃ）のパックドダブルワード（２個の３２ビットデータ）、図６（ｄ）のクワッドワード（１個の６４ビットデータ）の４種類のデータ形式が用意されている。

【０００４】このように、ＳＩＭＤ型プロセッサでは、レジスタに格納されるデータ形式が複数存在する。このため、このようなデータのＳＩＭＤ演算を行うために、ＳＩＭＤ型命令が各データタイプごとに用意されている。例えば、MMXテクノロジでは、加算を実行するために、図６（ａ）のパックドバイト、図６（ｂ）のパックドワード、図６（ｃ）のパックドダブルワードの３種類のデータタイプごとに、加算命令が用意されている。また、演算結果のアンダーフローおよびオーバーフローを防ぐサチュレーション付き加算なども含めると、加算命令だけで、７種類の命令が用意されている。さらに、加算だけでなく、減算、乗算、シフトなどの命令についても、データタイプごとに用意されている。

【０００５】このため、従来のＳＩＭＤ型プロセッサでは、ＳＩＭＤ型命令を実行するために、数多くの命令が定義されていた。

【０００６】

【発明が解決しようとする課題】しかしながら、命令数が増加すると、これらの命令を区別するためのオペランドに多くのビット数を割り当てる必要があることから、命令コード全体が増加し、プログラムのコード効率が低下するという問題があった。図７は、従来の命令コード

4

の一例を示す図である。

【０００７】図７において、命令コードは、例えば、３２ビットで構成され、この命令コードにより、オペランドＯＰ、出力レジスタｒｔおよび入力レジスタｒｓ、ｒｄを指定することができる。ここで、命令数が増加すると、オペランドＯＰに多くのビット数を必要とし、命令コード全体が増加する。

【０００８】一方、プログラムのコード効率の低下を防止するため、出力レジスタｒｔおよび入力レジスタｒｓ、ｒｄに割り当てるビット数を減らすことにより、オペランドＯＰのビット数の増加分を相殺する方法もあるが、この方法では、レジスタ選択の自由度が制約される。そこで、本発明の目的は、演算機能の多様性を損なうことなく、ＳＩＭＤ型命令の命令数を減らすことが可能なプロセッサ、データタイプ指定方法、データロード方法、演算方法、データタイプ更新方法および命令プログラムを提供することである。

【０００９】

【課題を解決するための手段】上述した課題を解決するために、請求項１記載のプロセッサによれば、データを記憶するデータメモリと、ロード命令で特定されるデータを前記データメモリからロードする入力レジスタと、前記ロード命令で特定されるデータのデータタイプを指定するデータタイプ指定手段と、前記データタイプ指定手段で指定されたデータタイプを、前記入力レジスタに対応させて記憶するデータタイプ記憶領域と、前記データタイプ記憶領域に記憶されたデータタイプに基づいて、前記入力レジスタにロードされたデータの演算を行う演算器とを備えることを特徴とする。

【００１０】これにより、ロード命令をデータタイプごとに用意するだけで、データタイプを区別することが可能となり、データタイプに対応した命令を演算種別ごとに用意する必要がなくなることから、ＳＩＭＤ型命令の命令数を減らすことが可能となる。さらに、演算器は、データタイプ記憶領域を参照するだけで、演算対象となるデータのデータタイプを判別することが可能となり、データタイプごとに異なる命令を与えることなく、データタイプに応じた演算を行うことが可能となることから、多種多様な演算を少ない命令数で行うことが可能となる。

【００１１】また、請求項２記載のプロセッサによれば、データを記憶するデータメモリと、ロード命令で特定されるデータを前記データメモリからロードする入力レジスタと、前記ロード命令で特定されるデータのデータタイプを指定するデータタイプ指定手段と、前記データタイプ指定手段で指定されたデータタイプを、前記入力レジスタに対応させて記憶する第１データタイプ記憶領域と、前記データタイプ記憶領域に記憶されたデータタイプに基づいて、前記入力レジスタにロードされたデータの演算を行う演算器と、前記演算器によるデータの

5

演算結果を記憶する出力レジスタと、前記データの演算結果に対応したデータタイプを、前記出力レジスタに対応させて記憶する第２データタイプ記憶領域とを備えることを特徴とする。

【００１２】これにより、演算結果に応じてデータタイプを更新することが可能となり、演算処理によりデータタイプが変わる場合においても、データタイプの整合性を維持することができ、演算命令が実行されるごとにデータタイプを指定し直す必要をなくすことができる。また、請求項３記載のプロセッサによれば、データタイプの異なるデータを記憶する入力レジスタと、前記データタイプの異なるデータの演算を行う演算手段と、前記演算手段による演算結果を記憶する出力レジスタとを備えることを特徴とする。

【００１３】これにより、データタイプの組み合わせに基づいて演算を定義することが可能となり、命令数を増加させることなく、ＳＩＭＤ演算のバリエーションを増やすことが可能となる。また、請求項４記載のデータタイプ指定方法によれば、ＳＩＭＤ型命令のデータタイプをロード命令により指定することを特徴とする。

【００１４】これにより、データをレジスタにロードする際に、データタイプを区別することが可能となり、データタイプに対応した命令を演算種別ごとに用意する必要がなくなることから、ＳＩＭＤ型命令の命令数を減らすことが可能となる。また、請求項５記載のデータタイプ指定方法によれば、前記ロード命令はデータタイプごとに用意されていることを特徴とする。

【００１５】これにより、データタイプに対応した命令を演算種別ごとに用意することなく、データタイプを区別することが可能となり、ＳＩＭＤ型命令の命令数を減らすことが可能となる。また、請求項６記載のデータロード方法によれば、ロード命令で特定されるデータをレジスタにロードするステップと、前記ロードされたデータに対応するデータタイプを、前記レジスタに対応させて記憶するステップとを備えることを特徴とする。

【００１６】これにより、レジスタにロードされたデータを取り込む際に、そのデータタイプも同時に取り込むことができ、単一の命令を与えるだけで、データタイプに応じた演算を行うことが可能となることから、ＳＩＭＤ型命令の命令数を減らすことが可能となる。また、請求項７記載の演算方法によれば、ロード命令で特定されるデータを入力レジスタにロードするステップと、前記ロードされたデータに対応するデータタイプを、前記入力レジスタに対応させて記憶するステップと、前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、前記データタイプに基づいて前記データの演算を行うステップとを備えることを特徴とする。

【００１７】これにより、ロード命令をデータタイプごとに用意するだけで、データタイプを区別することが可

6

能となり、ＳＩＭＤ型命令の命令数を減らすことが可能となるとともに、単一の命令を与えるだけで、データタイプに応じた演算を行うことが可能となることから、多種多様な演算を少ない命令数で行うことが可能となる。

【００１８】また、請求項８記載のデータタイプ更新方法によれば、ロード命令で特定されるデータを入力レジスタにロードするステップと、前記ロードされたデータに対応するデータタイプを、前記入力レジスタに対応させて記憶するステップと、前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、前記データタイプに基づいて前記データの演算を行うステップと、前記データの演算結果を出力レジスタに記憶するステップと、前記データの演算結果に対応したデータタイプを、前記出力レジスタに対応させて記憶するステップとを備えることを特徴とする。

【００１９】これにより、演算結果に応じてデータタイプを更新することが可能となり、演算処理によりデータタイプが変わる場合においても、データタイプの整合性を維持することができる。また、請求項９記載の命令プログラムによれば、ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、前記ロード命令で特定されるデータをレジスタにロードするステップと、前記ロード命令で指定されるデータタイプを、前記レジスタに対応させて記憶するステップとをコンピュータに実行させることを特徴とする。

【００２０】これにより、データをレジスタにロードする際に、データタイプを区別することが可能となり、データタイプに対応した命令を演算種別ごとに用意する必要がなくなることから、ＳＩＭＤ型命令の命令数を減らすことが可能となる。また、請求項１０記載の命令プログラムによれば、ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、前記ロード命令で特定されるデータをレジスタにロードするステップと、前記ロード命令で指定されるデータタイプを、前記レジスタに対応させて記憶するステップと、前記レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、前記データタイプに基づいて、前記演算器に演算処理を行わせるステップとをコンピュータに実行させることを特徴とする。

【００２１】これにより、ロード命令をデータタイプごとに用意するだけで、データタイプを区別することが可能となり、ＳＩＭＤ型命令の命令数を減らすことが可能となるとともに、単一の命令を与えるだけで、データタイプに応じた演算を行うことが可能となることから、多種多様な演算を少ない命令数で行うことが可能となる。

【００２２】また、請求項１１記載の命令プログラムによれば、ＳＩＭＤ型命令のデータタイプをロード命令により指定するステップと、前記ロード命令で特定されるデータを入力レジスタにロードするステップと、前記ロード命令で指定されるデータタイプを、前記入力レジス

7

タに対応させて記憶するステップと、前記入力レジスタにロードされたデータを、前記データタイプとともに演算器に入力するステップと、前記データタイプに基づいて、前記演算器に演算処理を行わせるステップと、前記演算処理結果を出力レジスタに記憶するステップと、前記演算処理結果に対応したデータタイプを、前記出力レジスタに対応させて記憶するステップとをコンピュータに実行させることを特徴とする。

【００２３】これにより、演算結果に応じてデータタイプを更新することが可能となり、演算処理によりデータタイプが変わる場合においても、データタイプの整合性を維持することができる。

【００２４】

【発明の実施の形態】以下、本発明の実施形態に係るプロセッサについて、図面を参照しながら説明する。図１は、本発明の第１実施形態に係るプロセッサの概略構成を示すブロック図である。

【００２５】図１において、データメモリ１にはデータが格納され、ロード命令に従って、データをレジスタファイル２にロードしたり、レジスタファイル２のデータを格納したりする。レジスタファイル２には、レジスタ２ａ～２ｃが設けられるとともに、データタイプ記憶領域３が設けられている。ここで、データタイプ記憶領域３は、レジスタ２ａ～２ｃごとに設けられ、これらのレジスタ２ａ～２ｃに対応して、データタイプ記憶領域３ａ～３ｃが設けられている。

【００２６】図２は、本発明の第１実施形態に係るレジスタ２ａ～２ｃの構成例を示す図である。図２において、レジスタ２ａ～２ｃには、例えば、３２ビットのデータ記憶領域１１が設けられ、データ記憶領域１１には、２ビット分のデータタイプ記憶領域１２が付加されている。

【００２７】ここで、データタイプとして、例えば、８ビットの４パックデータ、１６ビットの２パックデータ、３２ビットのデータ、浮動少数点形式データの４種類が設けられ、データタイプを識別するための識別データとして、８ビットの４パックデータには“００”、１６ビットの２パックデータには“０１”、３２ビットのデータには“１０”、浮動少数点形式データには“１１”が割り当てられている。

【００２８】そして、８ビットの４パックデータがデータ記憶領域１１に格納されると、データタイプ記憶領域１２には、そのデータ記憶領域１１に対応して“００”が格納され、１６ビットの２パックデータがデータ記憶領域１１に格納されると、データタイプ記憶領域１２には、そのデータ記憶領域１１に対応して“０１”が格納され、３２ビットのデータがデータ記憶領域１１に格納されると、データタイプ記憶領域１２には、そのデータ記憶領域１１に対応して“１０”が格納され、浮動少数点形式データがデータ記憶領域１１に格納されると、データタイプ記憶領域１２には、そのデータ記憶領域１１に対応して“１１”が格納される。

8

【００２９】ＳＩＭＤ演算器４は、レジスタファイル２、データタイプ記憶領域３および命令デコーダ５に接続されている。そして、レジスタファイル２に記憶されているデータを、データタイプ記憶領域３に記憶されているデータタイプとともに読み込み、そのデータタイプに応じた演算を行う。そして、その演算結果をレジスタファイル２に出力するとともに、その演算結果に対応したデータタイプをデータタイプ記憶領域３に出力する。

【００３０】命令デコーダ５は、プログラムメモリ６から読み出された命令コードに基づいて、レジスタファイル２、データタイプ記憶領域３およびＳＩＭＤ演算器４を制御する制御信号を発生する。プログラムメモリ６には、命令コードが格納され、命令コードを命令デコーダ５に出力する。ここで、ロード命令は、データタイプごとに用意される。

【００３１】図３は、本発明の第１実施形態に係る命令の一例を示す図である。図３において、ロード命令は、図２の８ビットの４パックデータ、１６ビットの２パックデータ、３２ビットのデータおよび浮動少数点形式データに対応して、４種類の命令ＬＤＰＢ、ＬＤＰＷ、ＬＤＤＷ、ＬＤＦＳが設けられる。また、加算や減算などの演算命令は、異なるデータタイプに共通に設けられる。

【００３２】次に、図１のプロセッサの動作について説明する。プログラムメモリ６からロード命令が読み出されると（①）、命令デコーダ５は、ロード命令を解読する。そして、データメモリ１から読み出されたデータを（④）、ロード命令で指定されたレジスタ２ａ～２ｃに記憶させる（②）。ここで、ロード命令は、データタイプごとに設けられ、命令デコーダ５は、ロード命令からデータタイプを判別する。そして、データが記憶されたレジスタ２ａ～２ｃに対応するデータタイプ記憶領域３ａ～３ｃに、そのデータタイプを記憶させる（③）。

【００３３】例えば、ロード命令のニーモニックが図３のＬＤＰＢであり、このロード命令により、レジスタファイル２中のレジスタ２ａが入力レジスタとして指定されているとすると、命令デコーダ５は、このニーモニックＬＤＰＢを解読することにより、ロード対象となるデータが図２の８ビットの４パックデータと判別し、そのデータタイプの識別データとして“００”を取得する。

【００３４】そして、命令デコーダ５は、この８ビットの４パックデータをデータメモリ１から読み出させ、レジスタファイル２のレジスタ２ａに記憶させるとともに、このレジスタ２ａに対応して設けられているデータタイプ記憶領域３ａに“００”を記憶させる。これにより、ロード命令をデータタイプごとに設けるだけで、レジスタ２ａ～２ｃに記憶されるデータのデータタイプを判別することが可能となる。

9

【００３５】次に、プログラムメモリ６から演算命令が読み出されると（⑤）、命令デコーダ５は、その演算命令を解読し、ＳＩＭＤ演算器４に制御信号を出力する（⑥）。ＳＩＭＤ演算器４は、命令デコーダ５から制御信号が出力されると、そこで指定されているレジスタ２ａ～２ｃからデータを取り込むとともに（⑦）、そのレジスタ２ａ～２ｃに対応するデータタイプ記憶領域３ａ～３ｃからデータタイプを取り込む（⑧）。そして、そのデータタイプに従ったデータのＳＩＭＤ演算を行い、指定されたレジスタ２ａ～２ｃにその演算結果を記憶させるとともに（⑨）、その出力先のレジスタ２ａ～２ｃに対応するデータタイプ記憶領域３ａ～３ｃにデータタイプを記憶させる（丸１０）。

【００３６】例えば、プログラムメモリ６から加算命令が読み出され、この加算命令で、レジスタファイル２中のレジスタ２ａ、２ｂが入力レジスタとして指定され、レジスタ２ｃが出力レジスタとして指定されているものとする。この場合、命令デコーダ５は、ＳＩＭＤ演算器４に加算指示を行うとともに、レジスタ２ａ、２ｂを入力レジスタとして指定し、レジスタ２ｃを出力レジスタとして指定する。

【００３７】ＳＩＭＤ演算器４は、レジスタ２ａ、２ｂが入力レジスタとして指定されると、レジスタ２ａ、２ｂからデータを読み込むとともに、そのレジスタ２ａ、２ｂに対応したデータタイプ記憶領域３ａ、３ｂからデータタイプを読み込む。そして、ＳＩＭＤ演算器４は、このデータタイプに従って、レジスタ２ａ、２ｂのデータを加算し、その加算結果をレジスタ２ｃに出力するとともに、そのレジスタ２ｃに対応したデータタイプ記憶領域３ｃにデータタイプを出力する。

【００３８】ここで、演算命令はデータタイプに共通に設けられ、命令デコーダ５は、プログラムメモリ６から演算命令が送られると、加算や乗算などの指示のみをＳＩＭＤ演算器４に送るとともに、入力レジスタと出力レジスタを指示する信号をレジスタファイル２に送る。そして、ＳＩＭＤ演算器４は、命令デコーダ５からの演算の指示に基づいて、演算対象となるデータのデータタイプを判断するのではなく、データタイプ記憶領域３ａ～３ｃに記憶されているデータに基づいて、演算対象となるデータのデータタイプを判断する。

【００３９】これにより、データタイプに対応した命令を、加算、減算、乗算などの演算種別ごとに用意する必要がなくなり、ＳＩＭＤ型命令の命令数を減らすことが可能となる。なお、データタイプ記憶領域３に記憶されるデータタイプは、ロード命令で指定するだけでなく、ＳＩＭＤ演算器４による演算結果に基づいて更新するようにしてもよい。

【００４０】図４は、本発明の第２実施形態に係るデータタイプの更新方法を示す図である。図４において、１６ビットの２パックデータａ０、ａ１がデータ記憶領域

10

１１ａに記憶されるとともに、データタイプ識別データ“０１”がデータタイプ記憶領域１２ａに記憶され、１６ビットの２パックデータｂ０、ｂ１がデータ記憶領域１１ｂに記憶されるとともに、データタイプ識別データ“０１”がデータタイプ記憶領域１２ｂに記憶され、これらのデータの乗算を行うものとする。

【００４１】この場合、ａ０とｂ０との乗算により３２ビットのデータが生成され、その３２ビットの乗算結果ａ０×ｂ０がデータ記憶領域１１ｃに記憶されるとともに、ａ１とｂ１との乗算により３２ビットのデータが生成され、その３２ビットの乗算結果ａ１×ｂ１がデータ記憶領域１１ｄに記憶される。ここで、データ記憶領域１１ａ、１１ｂに記憶されているデータは１６ビットの２パックデータ、データ記憶領域１１ｃ、１１ｄに記憶されるデータは３２ビットのデータであるため、演算前後でデータタイプが異なっている。

【００４２】このため、ＳＩＭＤ演算器４は、データタイプ記憶領域１２ａ、１２ｂに記憶されているデータタイプ識別データ“０１”を更新し、データタイプ記憶領域１２ｃ、１２ｄには、３２ビットのデータに対応したデータタイプ識別データ“１０”を記憶する。これにより、演算結果に応じてデータタイプが変わる場合においても、データタイプごとに異なった命令を与えることなく、データタイプの整合性を維持することが可能となり、データタイプ記憶領域１２ｃ、１２ｄを参照するだけで、その後の処理を適切に行うことが可能となる。

【００４３】図５は、本発明の第３実施形態に係る演算方法を示す図である。図５において、３２ビットのデータａがデータ記憶領域２１ａに記憶されるとともに、８ビットの４パックデータｂ０、ｂ１、ｂ２、ｂ３がデータ記憶領域２１ｂに記憶されている。ここで、３２ビットのデータａと８ビットの４パックデータｂ０、ｂ１、ｂ２、ｂ３との加算を、３２ビットのデータａのうちの８ビット分と８ビットの４パックデータｂ０、ｂ１、ｂ２、ｂ３との加算と定義し、これらのデータの加算を図１のＳＩＭＤ演算器４で行えるようにする。

【００４４】そして、ＳＩＭＤ演算器４に加算命令が出力されると、ＳＩＭＤ演算器４は、上記の定義に従って、データ記憶領域２１ａのデータａと、データ記憶領域２１ｂのデータｂ０、ｂ１、ｂ２、ｂ３との加算を行う。そして、これら異なるデータタイプ同士の加算結果として、ａ＋ｂ０、ａ＋ｂ１、ａ＋ｂ１、ａ＋ｂ１が得られると、これらの加算結果を８ビットの４パックデータとして、データ記憶領域２１ｃに記憶する。

【００４５】これにより、命令数を増やすことなく、演算のバリエションを増やすことができ、プログラムのコード効率を向上させることが可能となる。

【００４６】

【発明の効果】以上説明したように、本発明によれば、ロード命令をデータタイプごとに用意するだけで、デー

タタイプを区別することが可能となり、演算処理の多様性を損なうことなく、ＳＩＭＤ型命令の命令数を減らすことが可能となる。

【図面の簡単な説明】

【図１】本発明の第１実施形態に係るプロセッサの概略構成を示すブロック図である。

【図２】本発明の第１実施形態に係るレジスタの構成例を示す図である。

【図３】本発明の第１実施形態に係る命令の一例を示す図である。

【図４】本発明の第２実施形態に係るデータタイプの更新方法を示す図である。

【図５】本発明の第３実施形態に係る演算方法を示す図である。

【図６】従来のMMXテクノロジのデータタイプを示す図である。

【図７】従来の命令コードの一例を示す図である。

【符号の説明】

1　データメモリ

2　レジスタファイル

２ａ～２ｃ　レジスタ

3　データタイプ

３ａ～３ｃ、１２　データタイプ記憶領域

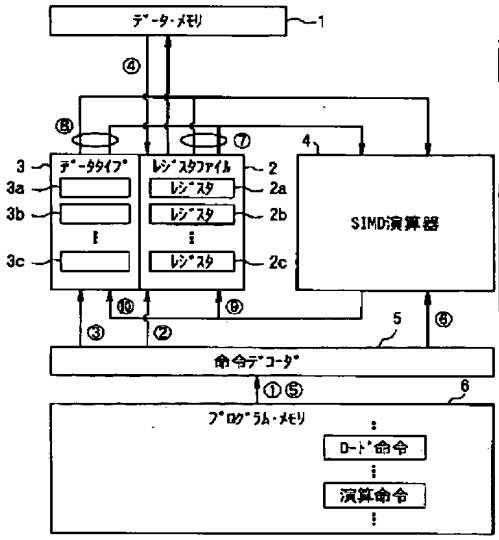
4　ＳＩＭＤ演算器

5　命令デコーダ

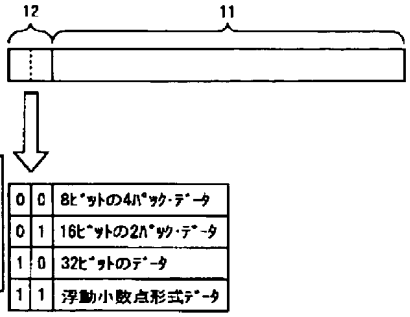
6　プログラムメモリ

１１　データ記憶領域

【図１】



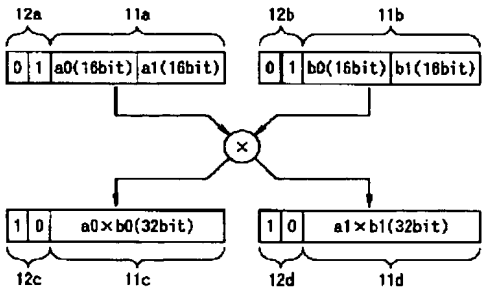
【図２】



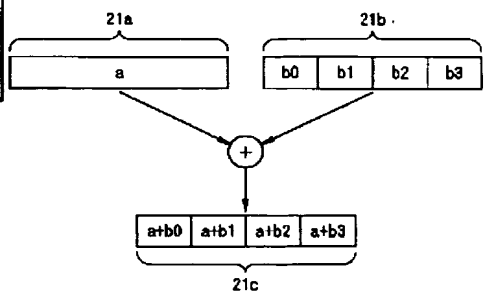
【図３】

機能	ニーモニック
ロード	LDPB,LDPW,LDDW,LDFS
加算	ADD
減算	SUB
サチュレーション付き加算	ADDS
サチュレーション付き減算	SUBS
⋮	⋮

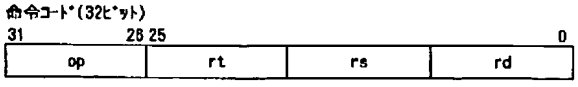
【図４】



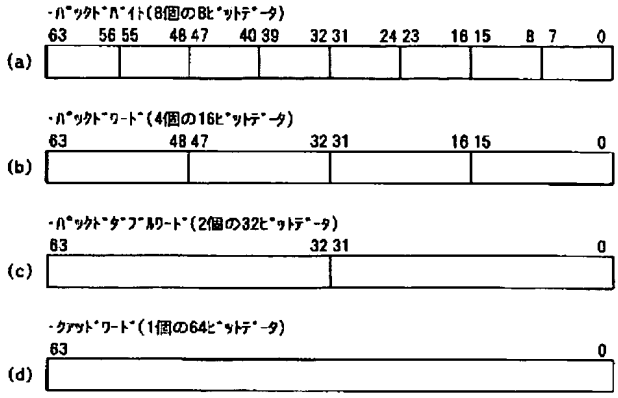
【図５】



【図７】



【図６】


・ﾊﾟｯｸﾄﾞﾊﾞｲﾄ(8個の8ﾋﾞｯﾄﾃﾞｰﾀ)
63 56 55 48 47 40 39 32 31 24 23 16 15 8 7 0
(a)
・ﾊﾟｯｸﾄﾞﾜｰﾄﾞ(4個の16ﾋﾞｯﾄﾃﾞｰﾀ)
63 48 47 32 31 16 15 0
(b)
・ﾊﾟｯｸﾄﾞﾀﾞﾌﾞﾙﾜｰﾄﾞ(2個の32ﾋﾞｯﾄﾃﾞｰﾀ)
63 32 31 0
(c)
・ｸｱｯﾄﾞﾜｰﾄﾞ(1個の64ﾋﾞｯﾄﾃﾞｰﾀ)
63 0
(d)